Joint Source-Channel Trellis Coding using Convolutional Codes' Partial Transfer Function and the BCJR Algorithm

Emmanuel Boutillon* and Luis González-Pérez**

*L.E.S.T.E.R. Centre de Recherche Université de Bretagne Sud 56325 LORIENT Cedex FRANCE
**ENST Paris; 46, rue Barrault, 75634 Paris Cedex 13, FRANCE
E-mail: bout@enst.fr, Luis.Gonzalez@enst.fr

Abstract: *A Joint Source-Channel Trellis Coding technique consisting in the joint optimizaton of a trellis source coder and a convolutional code is presented. The pairwise error probability needed for the codebook design and the encoding processes is analytically estimated with the transfer function of the convolutional code. This way, no extensive monte-carlo simulations are required. In addition, at the receiver end the BCJR algorithm is modified to act as a "soft source decoder". By using the channel estimation of the BCJR algorithm, the distortion introduced by channel errors can be further reduced.*

Keywords: JSCC, trellis quantization, BCJR, MAP, robust codebook design.

1. INTRODUCTION

The Joint Source-Channel Trellis Coding (JSCTC) system performs a Trellis Quantization (TQ) of the source "robust" to transmission errors. In this scheme, errors are not corrected but their contribution to the distortion of the reconstructed signal is minimized. The codebook design and the encoding process are both optimized to minimize the expectation of the distortion between the original message $\{x_t\}_{t=0}^{L-1}$ and the one reconstructed at the receiver end $\{\hat{x}_t\}_{t=0}^{L-1}$

$$D = \sum_{t=0}^{L-1} (x_t - \hat{x}_t)^2 \qquad (1)$$

Let us define $C = \{y_i\}_{i=0}^{2^K-1}$ the reproduction codebook, where y_i is the codeword associated to the i^{th} branch of the trellis and K is the constraint length of the encoder. It is shown in [4] that the optimal encoding of the source can be done by using the Viterbi Algorithm (VA) with the branch metric

$$\begin{aligned} d(x_t, y_j) &= E\{(x_t - y_i)^2 | y_j)\} \quad i = 0 \ldots 2^K - 1 \\ &= \sum_{i=0}^{2^K-1} (x - y_i)^2 \; P(y_i|y_j) \end{aligned} \qquad (2)$$

where $P(y_i|y_j)$ is the pairwise error probability (PEP) of decoding codeword y_i given that codeword y_j was sent. Note that if x_t is quantized with y_i, expression (2) is the elementary contribution of the t^{th} symbol to the overall expectation of the distortion.

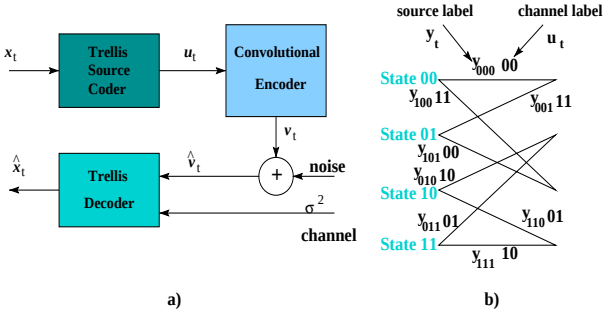

Figure 1: Joint Source-Channel Trellis Coder

In the simplest case, i.e. a TQ that generates the binary sequence $\{u_t\}$ (i.e. the optimal path) and sends it through a binary symmetric channel (BSC), the analytical value of $P(y_i|y_j)$ is given by the simple function

$$P(y_i|y_j) = p^{d_H(i,j)} \cdot (1-p)^{K - d_H(i,j)} \qquad (3)$$

where $d_H(i,j)$ denotes the Hamming distance between the channel labels of the reproduction codewords y_i and y_j and p is the channel transition probability. Equation (3) gives the probability of sending bits $(u_t, \cdots, u_{t-K+1})$ that define the binary value of codeword y_j and receiving $(\hat{u}_t, \cdots, \hat{u}_{t-K+1})$, the binary representation of y_i.

When the TQ is concatenated with a convolutional encoder of the same constraint length (see fig. 1.a), the channel symbols $\{v_t\}$ associated to the optimal path are sent to the receiver instead of $\{u_t\}$. From the received symbols $\{\hat{v}_t\}$ corrupted by noise, the trellis decoder decodes $\{\hat{x}_t\}$. In this particular case, the computaion of the PEPs, as well as the codebook design problem become far more complex. All the reported work related to this problem proposes an indirect estimation of the PEP through the use of real channel measurements or Monte-Carlo simulations [1], [2].

This paper explores the utilization of restricted versions of the "two-way type" algorithm to extract information at a branch level rather than at a bit level. For the encoding process, the transfer function of the convolutional code is used to obtain its

distance properties with respect to a specific branch in the trellis. The weight distribution for this specific branch allows an accurate estimation of the PEP, and thus, a more efficient source quantization and codebook design. At the encoder end, the use of a restricted version of the BCJR algorithm (also known as MAP or Forward-Backward algorithm) is proposed to obtain the a posteriori probability (APP) of all the branches. This information is used to perform "soft source decoding", i.e. to reconstruct the optimum reproduction sample.

Section 2 reminds the principles of JSCTC. Then, sections 3, 4 and 5 present the estimation of the PEP, the compression-extension codebook generation algorihm and the soft source decoding, respectively. Finally, section 6 gives simulation results. A comparison with non jointly optimized systems is also done.

2. PEP COMPUTATION

As stated above, in the proposed scheme, the error probability depends on the channel code. The usual approaches to deal with this problem are based on monte-carlo simulations [1], [2]. Unfortunately, those methods are not very accurate and they are time consumming.

Since the trellises for the source and channel coders are the same, the probability $P(y_i|y_0)$ is equal to the probability that the VA decodes the channel symbol corresponding to branch i instead of decoding the channel symbol corresponding to branch number 0. This probability depends on the code generator polynomial, i.e. the weight distribution of all the paths in the trellis. In [5], a similar problem is developed to obtain an upper bound of the first event error probability. The same technique is employed in this paper. $P(y_i|y_j)$ can be upper- bounded with

$$P(y_i|y_0) \leq \Sigma_{d=d_{free}}^{\infty} a_i(d)\ P_b(d) \tag{4}$$

where d_{free} is the free distance of the convolutional code and coefficient $a_i(d)$ denotes the number of trellis paths of weight d that diverge from the all-zero path, pass by the trellis branch y_i and remerge into the all-zero path. $P_b(d)$ is the probability of error in a pairwise comparison of two paths that differ in d bits [5]

$$P_b(d) = Q\left(\sqrt{2\frac{E_b}{N_o}R_c\ d}\right) \tag{5}$$

where R_c is the channel code rate.

Since the channel code is linear, equation (4), which was obtained for the all zero path, can be generalized to all trellis paths and PEPs ($P(y_i|y_j) = P(y_{i\oplus j}|y_0)$).

Coefficients $a_i(d)$ can be obtained analytically. Proceeding in the same manner as in [5] for the computation of the convolutional code's tranfer function $T(D)$, a label y_i is added to branch i of the code's state diagram so as to obtain a modified tranfer function $T(D, y_i)$. Coefficient $a_i(d)$ is given by the coefficients of polynomial

$$P_i(D) = \left.\frac{\partial T(D, y_i)}{\partial y_i}\right|_{y_i=1} \tag{6}$$

In practice, we have obtained the first coefficients of polynomial $P_i(D)$ using a partial version (L iteration) of the two-way algorithm described in [6]. Let m' and m be two states of the trellis and define the branch tranfer polynomial $G(m', m)$ as

$$G(m', m) = \delta(m', m) \cdot D^{d_H(m', m)} \tag{7}$$

where $\delta(m', m)$ is equal to 0 if node m' and m are not connected, 1 otherwise; and $d_H(m', m)$ is the number of bits differing in the channel symbols associated to branch 0 and (m', m). In order to avoid unwanted reconvergence to the all zero path, branch $(m', 0)$ and $(0, m)$ are deleted respectively for the forward and the backward recursion

$$G_F(m', m) = Z(m) \cdot G(m', m) \tag{8}$$

$$G_B(m', m) = Z(m') \cdot G(m', m) \tag{9}$$

where $Z(x)$ equals 0 if $x = 0$, 1 otherwise.

The forward recursion computes recursively, from $t = 1$ to L, the polynomial $A_t(m')$ that gives all the paths of length smaller of equal to t leaving the all zero path and leading to node m'. The initial conditions (path of length 0) are $A_0(m') = 1 - Z(m')$.

$$A_t(m) = \sum_{m'} A_{t-1}(m') \cdot G_F(m', m) \tag{10}$$

Symmetrically, the backward recursion computes $B_{2L+1-t}(m)$, the polynomial that gives all the paths of length smaller of equal to t that leave the node m and lead to the all zero path. The initial conditions are $B_{2L+1}(m) = 1 - Z(m')$.

$$B_{2L+1-t}(m') = \sum_{m} B_{2L+2-t}(m) \cdot G_B(m', m) \tag{11}$$

Merging $A_L(m')$ and $B_{L+1}(m)$ with the $G(m', m)$ gives the first L terms of polynomial enumerator $P_i(D)$, where i is associated with the branch (m, m')

$$P_i(D) = A_L(m') \cdot G(m', m) \cdot B_{L+1}(m) \tag{12}$$

Table 1 shows the PEP computation results for a $1/2$ rate, $K = 7$ convolutional code with polynomial

Table 1: Simulated and Estimated PEPs (10^{-4})

branch i	1	32	64	96	127
$P_s(y_i\|y_0)$	6.7	0.56	1.2	0.71	0.19
$P_{ub}(y_i\|y_0)$	11.6	1.0	1.6	0.72	0.20
$P_n(y_i\|y_0)$	7.3	0.65	1.0	0.45	0.13

generators $(133, 171)_{octal}$ and for SNR = 3 dB. The first line (P_s) gives a PEP estimation obtained with a Monte-carlo simulation (an all zero path of length 10^6). The second line (P_{ub}) gives the upper bound of the PEP obtained with (4). In order to obtain a sum of the estimated PEP equal to 1, the (P_{ub}) are normalized using $P_s(y_0|y_0)$

$$P_n(y_j|y_0) = P_{ub}(y_j|y_0) \cdot \frac{1 - P_s(y_0|y_0)}{\sum_{m>0} P_{ub}(y_m|y_0)} \qquad (13)$$

As shown in table 1, there is a good coincidence between the order of simulated and estimated values (the differences may come from the insufficient number of samples obtained by simulation).

From the estimated value of PEP, quasi optimal quantization can be performed. The next problem is to find an efficient codebook for the quantization.

3. CODEBOOK DESIGN

In [4], a codebook design algorithm for JSCTC was proposed. This algorithm is an iterative process consisting in two main steps:

- Encoding: for the codebook C^m of iteration m, perform the quantization of the trainning sequence.
- Codebook update: given the obtained quantization, compute C^{m+1} that minimizes the overall distortion.

This process is iterated until no significant distorsion improvement is obtained. In that case codebook C^f is the final optimized codebook. Notice that all those operations imply the knowledge of all PEPs $P(y_i|y_j)$.

This algorithm is efficient but very sensitive to the seed codebook C^0. Ayanoğlu et al. [4] propose an incremental codebook design. The idea is to start from a $K-1$-constraint length encoder and employ the codebook design algorithm to obtain an optimized codebook C^f_{K-1} for this encoder; then, codebook C^f_{K-1} is extended to a K-constraint length encoder by appending a register element at the LSB position. To do so, codeword y_i associated to branch i in C^f_{K-1} is associated to branches $2i$ and $2i+1$ of codebook C^0_K. Then, the codebook desing algorithm is used to obtain the final codebook C^f_K. This method is recursively applied from C^0_0 (a unique codeword) to C^f_K.

When a convolutional code is used, this method has to be adapted. Indeed, the PEPs used in the optimization process of the 2^K-size codebook have to be reduced for the optimization of the 2^{K-1}-size codebook. To do so, a reduction of the trellis is done by using

$$P^{k-1}(Y_i|y_0) = P^k(y_{2i}|y_0) + P^k(y_{2i+1}|y_0) \qquad (14)$$

In pratice, trellis reduction is first performed from $k = K$ to $k = 0$, then trellis extension is used from $k = 0$ to $k = K$. So far, the results of this algorithm have not been compared with other algorithms.

4. SOFT SOURCE DECODING

At the receiver end, the trellis decoder reconstruct the $\{\hat{x}_t\}$ from the $\{\hat{v}_t\}$. The method generally used is based on the concatenation of the VA that gives the decoded sequence $\{\hat{u}_t\}$ and a source decoder (K-length shift register addressing a 2^K-word Look-Up Table) that decodes $\{\hat{x}_t\} = \{\hat{y}_t\}$ from $\{\hat{u}_t\}$. In this paper we propose to merge the two operations to obtain a soft source decoding that minimizes the expectation of the distortion. Indeed, equation (2) can be bounded with

$$D \leq \sum_{t=0}^{L-1} (x_t - y_{i(t)})^2 + \sum_{t=0}^{L-1} (y_{i(t)} - \hat{x}_t)^2 \qquad (15)$$

The first term of (15) is the quantization distortion and the second term is the distortion introduced by transmission errors. The expectation of this term can be minimized using the weighted centroid (proof in [7])

$$\hat{x}_t = \frac{1}{\sum_{i=0}^{2^K-1} Pr_t(y_i)} \sum_{i=0}^{2^K-1} y_i Pr_t(y_i) \qquad (16)$$

where $Pr_t(y_i)$ is the APP, given all the received noisy signal $\{\hat{v}_t\}_{t=0}^{L-1}$, that the branch (m', m) labeling the source codeword y_i, was taken by encoder at time t. The $Pr_t(y_i)$ can be obtained directly by a restricted BCJR algorithm since those informations are naturally generated by this algorithm. Putting the probability $Pr_t(y_i)$ in terms of the notation employed in [3], we have

$$Pr_t(y_i) = Pr_t(y_{(m',m)}) = \sigma_t(m', m) \qquad (17)$$

Hence, equation (16) can be rewritten in the form[1].

[1] An independent work was recently published for the case of a TCQ encoder jointly optimized with a TCM coder [2].

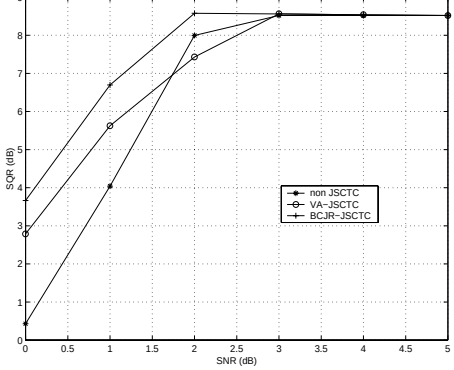

Figure 2: Source performace for the transmission of the first order Gauss-Markov source.

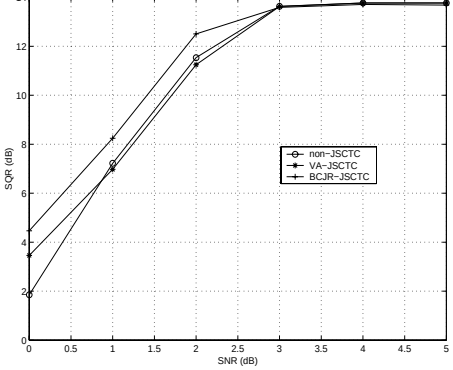

Figure 3: System performance for the transmission of the Lenna image.

$$\hat{x}_t = \frac{\sum_{m'} \sum_m y_{(m',m)} \sigma_t(m',m)}{\sum_{m'} \sum_m \sigma_t(m',m)} \qquad (18)$$

Using (18), the second term of equation (15) is minimized.

5. SIMULATION RESULTS

Preliminary simulation results are given for different configurations: simple case (non-JSCTC) composed of an independent TQ and convolutional encoder, an intermediate system usign results of section 2 an 3 (VA-JSCTC) and the proposed system using soft output decisions (BCJR-JSCTC). All simulations are performed with a $K = 7$, $R-1/2$ convolutional encoder with generator polynomials $(133, 171)_{octal}$. The TQ generates 0.5 bits per input symbol (y_i are thus coupled) in order to have an overall 1 bit per input symbol.

Figure 2 shows the results of a $\rho = 0.9$ Gauss-Markov source and figure 3 of a 512×512, 8-bit per pixel Lenna image. From the two figures one can notice that VA-JSCTC is above non-JSCTC for low SNR(+2dB and +1.5dB of signal to quantization noise ratio (SQR) at SNR of 0dB. The mitigate result for medium SNR may result of an inappropiate codebook design. Further simulations will be done, using different codebook generation algorithms, in order to test this hypothesis. In all cases, the use of the soft source decoding improve the SQR of 1 dB for low SNR.

6. CONCLUSION

A jointly optimized trellis source coder and convolutional code was presented. It was shown that the channel estimation for source encoding and codebook design can be tighly bounded with the transfer function of the convolutional code without the need of monte-carlo simulations. On the other hand, the BCJR algorithm was modified to perform "soft source decoding" with the use of its a posteriori probability estimation. Finally, significant performance improvement is achieved compared to separately designed source and channel coders, specially at low channel SNR.

REFERENCES

[1] Min Wang and T. R. Fischer, "Trellis-Coded Quantization Designed for Noisy Channels", *IEEE Trans. Inform. Theory*, vol. IT-40, pp. 1792-1802, Nov. 1994.

[2] K.-H. Chei and K.-P. Ho, "Design of Optimal Soft Decoding for Combined Trellis Coded Quantization/Modulation in Rayleigh Fading Channel", *ICASSP'00*, June 2000, Turkey.

[3] L. R. Bahl, J. Cocke, F. Jelinek and J. Raviv, "Optimal decoding of linear codes for minimizing symbol error rate", *IEEE Trans. Inform. Theory*, vol. IT-20, pp. 284-287, March 1974.

[4] E. Ayanoglu and R. M. Gray, "The Design of Joint Source and Channel Trellis Waveform Coders", *IEEE Trans. Inform. Theory*, vol. IT-33, pp. 855-865, Nov. 1987.

[5] J. G. Proakis, *Digital Communications*, McGraw-Hill, 1995.

[6] G.D. Forney, "On Iterative Decoding and the Two-Way algorithm", *Proceedings of the Int. Symp. on Turbo-Codes*, Sept. 97, Brest, France.

[7] S.B. Zahir Azami, *Codage conjoint source/canal, Protection hierarchique*, PhD. Disseration, ENST, France 1999.